On Outlier Exposure with Generative Models

Konstantin Kirchheim
Otto-von-Guericke University
Magdeburg, Germany
konstantin.kirchheim@ovgu.de

Frank Ortmeier
Otto-von-Guericke University
Magdeburg, Germany
frank.ortmeier@ovgu.de

Abstract

While Outlier Exposure reliably increases the performance of Out-of-Distribution detectors, it requires a set of available outliers. In this paper, we propose Generative Outlier Exposure (GOE), which alleviates the need for available outliers by using generative models to sample synthetic outliers from low-density regions of the data distribution. The approach requires no modification of the generator, works on image and text data, and can be used with pre-trained models. We demonstrate the effectiveness of generated outliers on several image and text datasets, including ImageNet.

1 Introduction

Out-of-Distribution (OOD) detection refers to the identification of data points with a low probability under the data generating distribution. Outlier Exposure (OE) [12], which involves training on a set of available example outliers, effectively increases the performance of OOD detectors. The general formulation of the optimization objective of OE is

$$\min_{\theta} \mathbb{E}_{(x,y)\sim p_{in}}[\mathcal{L}(x,y) + \lambda \mathbb{E}_{x'\sim p_{out}^{train}}[\mathcal{L}'(x,y,x')]] \tag{1}$$

where p_{in} is the training data distribution, p_{out}^{train} is some OOD data distribution, and the $\mathcal{L}$ are some loss functions. In the typical setting, a dataset unrelated to the original task is used as p_{out}^{train}. However, the availability of a sufficiently large dataset of representative outliers can be considered a strong assumption that might not hold in some settings.

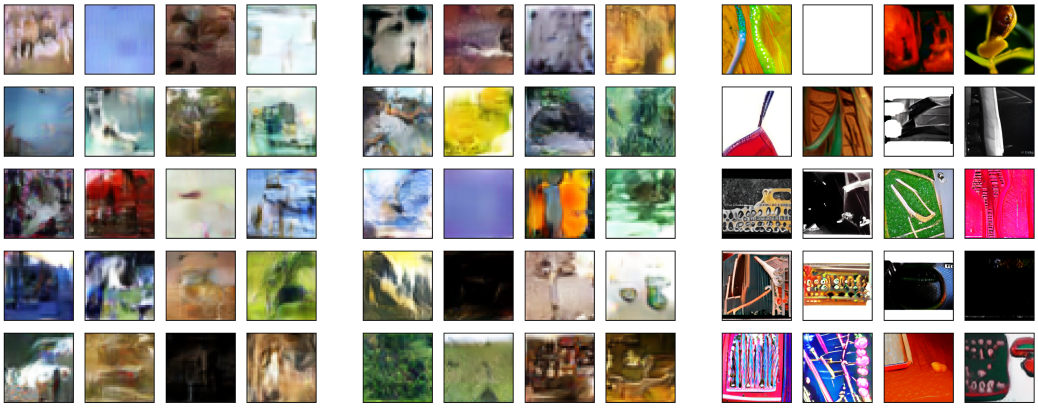
In this work, we explore the use of (possibly pre-trained) generative models to synthesize outliers. The proposed approach alleviates the need for available outliers by parameterizing p_{out}^{train} with a generative model from which points in low-density regions of the data distribution are drawn. We empirically demonstrate that such points can be used as representative example outliers such that models exposed to them generalize to OOD data from different distributions.

2 Synthesizing Outliers

Instead of replacing p_{out}^{train} by an existing dataset, we could learn a generative model p_G that samples from the low-density regions of p_{in}. Intuitively, such samples are closer to the decision boundary than outliers drawn from, say, a uniform distribution, and would therefore have a stronger regularizing effect on the OOD detector exposed to such outliers. There are at least two strategies to sample such points from generative models:

1. Biased generator: assuming a sufficient divergence between p_G and p_{in}, we can directly use the samples from the generator as outliers.
2. Sampling-parameters: some generators allow to control the variety of their output by adjusting sampling-parameters, such as temperature or variance. These parameters can be modified to sample low probability data points from unbiased generators.

NeurIPS ML Safety Workshop, 36th Conference on Neural Information Processing Systems (2022).



(a) CIFAR 10 $\sigma^2 = 50$ (b) CIFAR 100 $\sigma^2 = 50$ (c) ImageNet $\sigma^2 = 2$

Figure 1: Outliers generated by a BigGAN [2] at high variance σ^2. While the images do not depict recognizable objects, they still exhibit patterns found in naturally occurring data. Exposing models to these samples during training increases their OOD robustness over models exposed to uniform noise.

Using one of both approaches, we can sample outliers that still exhibit some of the patterns of natural data, which makes them efficient for Outlier Exposure. We will refer to this approach as Generative Outlier Exposure (GOE).

3 Experiments

We conduct experiments on common image and text classification datasets. For each task, we provide results for Maximum Softmax Probability (MSP) [10], as well as Energy-Based Out-of-Distribution Detection [19]. In addition to the baseline models, we compare the performance of models trained with Outlier Exposure using (1) generated outliers, (2) outliers sampled uniformly from the input space, and (3) natural outliers as originally proposed. Our implementation is based on PyTorch [21] and PyTorch-OOD [15], and publicly available.[1] Detailed information on the used hyper-parameters is provided in Appendix A.

3.1 Images

For CIFAR 10 and CIFAR 100, we train a BigGAN [2] as (class conditional) generative model $p_G(x'|y)$, where y is some class label. To sample outliers, we increase the variance σ^2 of the isotropic Gaussian $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ from which the latent vectors are drawn to 50. We generate 50,000 outlier images in total, sampling from all classes uniformly. A random selection is depicted in Fig. 1.

Following [10, 19, 17, 18], we use a WideResNet with 40 layers as our OOD detector backbone. During the evaluation, we measure the ability of the detectors to discriminate between samples from the corresponding test-set and several commonly used OOD datasets: Textures, LSUN Crop, LSUN Resize, TinyImageNet Crop, TinyImageNet Resize [18]. We calculate the performance for each OOD dataset individually and report the mean. Results can be found in Tab. 1.

For experiments on ImageNet-1K, we use a pre-trained BigGAN. Outliers are generated by sampling uniformly from all 1000 classes with a variance $\sigma^2 = 2$. As OOD detector backbone, we use a VisionTransformer (ViT) [4] pre-trained on the ImageNet-21K and 1K and fine-tune it with Outlier Exposure for 1000 batches of size 128. We use the ImageNet validation set as in-distribution, and ImageNet-A [13], ImageNet-R [9], and ImageNet-O [9] as OOD data during testing. Averaged performance measures can be found in Tab. 1.

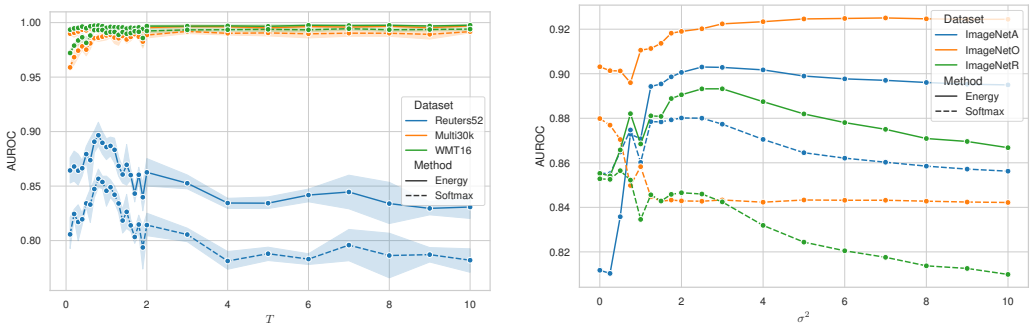
Overall, our method achieves competitive performance improvements over the baseline models. For the ImageNet dataset, where, to our knowledge, no commonly used Outlier Exposure data is available, GOE increases the performance significantly over models exposed to uniformly sampled outliers.

[1] `https://gitlab.com/kkirchheim/mlsafety-workshop-goe`

Table 1: Performance of OOD detection methods on image data averaged over different outlier datasets. ↑ indicates that larger values are better, while ↓ indicates the opposite – all values in percent.

Outlier	Method	AUROC ↑	AUPR-IN ↑	AUPR-OUT ↑	FPR@95TPR ↓
CIFAR 10					
–	MSP	91.74	88.42	93.47	28.94
–	Energy	93.03	91.43	93.70	31.57
Noise	MSP	94.49	91.75	96.11	17.83
Noise	Energy	97.36	96.32	97.95	11.54
Generated	MSP	95.54	94.20	96.54	18.31
Generated	Energy	95.51	94.04	96.52	18.50
Tiny300k	MSP	98.56	97.86	98.92	6.12
Tiny300k	Energy	**98.63**	**97.91**	**98.99**	**5.88**
CIFAR 100					
–	MSP	78.71	73.35	82.58	58.58
–	Energy	84.41	79.72	86.85	48.55
Noise	MSP	80.75	76.22	84.69	53.76
Noise	Energy	91.57	89.20	93.17	33.05
Generated	MSP	91.61	89.05	93.33	33.43
Generated	Energy	**92.68**	**89.72**	**94.48**	**26.26**
Tiny300k	MSP	87.13	81.49	89.97	37.91
Tiny300k	Energy	86.29	78.60	89.69	37.18
ImageNet 1K					
–	MSP	82.48	50.38	93.92	61.34
–	Energy	88.71	61.91	95.69	47.01
Noise	MSP	82.78	50.93	93.89	60.63
Noise	Energy	88.87	62.66	95.58	46.05
Generated	MSP	85.65	53.57	95.33	55.31
Generated	Energy	**90.34**	**62.16**	**96.76**	**39.47**

GOE results at different variance levels are depicted in Fig. 2. Training with $\sigma^2 = 0$ has a negligible effect, small σ^2 can have a detrimental effect, and larger σ^2 tend to increase the performance, where the improvement gradually decays with increasing variance.



(a) NewsGroups

(b) ImageNet

Figure 2: Performance of methods at different levels of variance σ^2 and temperature T. For NewsGroups, we calculate confidence intervals based on five experiments with different random seeds. We observe that low variance can be detrimental to the training, while medium levels of variance can improve the OOD robustness of models.

Table 2: Performance of OOD detection methods on text data averaged over three different outlier datasets. ↑ indicates that larger values are better, while ↓ indicates the opposite – all values in percent.

Outlier	Method	AUROC ↑	AUPR-IN ↑	AUPR-OUT ↑	FPR@95TPR ↓
–	Energy	89.35	69.95	94.56	31.93
–	MSP	82.06	62.08	88.74	52.02
Noise	Energy	93.93	78.20	97.02	18.94
Noise	MSP	93.22	77.65	96.42	23.11
Generated	MSP	93.02	84.92	96.89	22.55
Generated	Energy	**95.11**	**87.12**	**98.03**	17.50
Wiki2	Energy	94.41	86.23	97.87	**16.85**
Wiki2	MSP	93.84	86.34	97.56	19.55

3.2 Texts

We conduct experiments on text data based on the 20 NewsGroups dataset. We use a Transformer [22] as generator, while the OOD detection backbone is based on Gated Recurrent Units (GRU), similar to the one used by Hendrycks et al. [10].

We train the generator on the NewsGroups dataset and subsequently sample 100.000 words at temperature $T = 2$, which we use as outlier data during training. For comparison, we create synthetic texts by sampling random sequences of 20 tokens and use the Wiki2 dataset as natural outliers. We test all models against the Reuters52, Multi30k, and WMT16 Sentences datasets.

The averaged results are listed in Tab. 2. We observe that GOE outperforms the baselines, as well as models exposed to random token sequences (Noise). Models exposed to generated outliers perform on par or even outperform models exposed to natural outliers.

4 Related Work

Several previous works employed generative models for OOD detection. Virtual Outlier Synthesis [5] generates outliers in the feature space of a neural network by modeling the feature distribution with Gaussians. GOE instead models the outliers directly in the input space. Generative OpenMax [7] uses GANs to create outliers to improve the performance of the OpenMax Layer [1]. GOE, on the other hand, is layer-agnostic. AnoGAN [20] identifies OOD inputs by finding a point in the latent space of a generator that creates a point similar to the given input, which requires solving an optimization problem by back-propagation. The likelihood of the input is then estimated by evaluating the likelihood of the corresponding point in the latent space. GOE is computationally lightweight and does not introduce a computational overhead during inference. [16] proposes a training scheme that jointly trains a GAN to create outliers during training. GOE does not require joint training and can use pre-trained generative models.

5 Conclusion & Future Work

We presented Generative Outlier Exposure, a method that uses generative models to sample from low-density regions of the data distribution to generate synthetic Outlier Exposure data. We demonstrate that the proposed method outperforms Outlier Exposure with uniformly sampled outliers.

While we conducted experiments on image and text data, the concept constitutes a general framework that could be easily extended to other data modalities. Generative Outlier Exposure could also be combined with different variants of Outlier Exposure, such as the Objectosphere Loss [3] or the Energy-bounded Learning Loss [19]. Since the method can be used with off-the-shelf generative models and does not require retraining the generator, it provides a computationally cheap method for settings in which pre-trained generative models exist, but no outliers are available.

We have shown that good sampling-parameters are crucial to achieving competitive performance improvements. While we selected variance and temperature by manually inspecting the generated outliers, the automatic selection of adequate parameters remains an open problem.

References

[1] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1563–1572, 2016.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[3] Akshay Raj Dhamija, Manuel Günther, and Terrance Boult. Reducing network agnostophobia. In *Advances in Neural Information Processing Systems*, pages 9157–9168, 2018.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*, 2021.

[5] Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. VOS: Learning what you don't know by virtual outlier synthesis. In *International Conference on Learning Representations*, 2021.

[6] Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems*, 34, 2021.

[7] Zongyuan Ge, Sergey Demyanov, Zetao Chen, and Rahil Garnavi. Generative openmax for multi-class open set classification. In *British Machine Vision Conference 2017*. British Machine Vision Association and Society for Pattern Recognition, 2017.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[10] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *International Conference on Learning Representations*, 2017.

[11] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR, 2019.

[12] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2018.

[13] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[15] Konstantin Kirchheim, Marco Filax, and Frank Ortmeier. PyTorch-OOD: A library for out-of-distribution detection based on pytorch. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4351–4360, June 2022.

[16] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*, 2018.

[17] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in Neural Information Processing Systems*, 31, 2018.

[18] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[19] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33, 2020.

[20] Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual images. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 620–635, Cham, 2018. Springer International Publishing.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

A Hyperparameters

Text As generative model, we use the transformer architecture from the official PyTorch examples[2] without modifications, and with the default hyperparameters.

We train the GRU for 10 epochs, using the Adam optimizer [14] with a batch size of 32. For Outlier Exposure, we use $\lambda = 0.5$.

Image To train the BigGAN, we used the training scripts provided by the authors with the default hyperparameters.[3]

We trained all classification models with SGD, with a learning rate of 0.001. For the CIFAR datasets, we used the pre-trained models provided by [12] and fine-tuned them with Outlier Exposure with a batch size of 256, using $\lambda = 0.5$, for 25 epochs. Our ViT (base) operates on images of size 224×224, with a patch size of 16.

B Ablation Studies

Model Architecture In addition to the results for the ViT, we conduct experiments with a ResNet-101 [8] on the ImageNet. Quantitative performance measures are given in Tab. 3. We observe results similar to the transformer model; however, the overall performance is slightly decreased. The main contributing factor behind this difference is the increased performance of the ViT on the ImageNet-O, which, as we hypothesize, is due to the following:

- Pre-training on large-scale datasets was shown to improve model uncertainty [11, 6].
- Attention mechanisms were shown to increase the performance of OOD detectors, presumably due to their ability to model global structures [13].

Results for different levels of variance of the generator are depicted in Fig. 3. Again, the results are similar to the ones observed for the Transformer-based model.

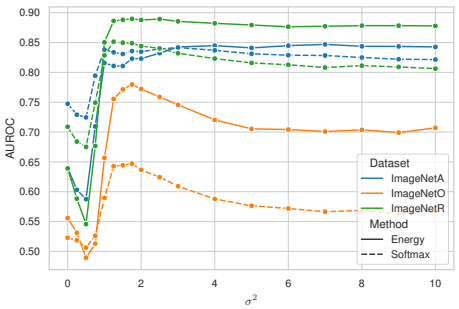


Figure 3: Influence of generator variance on a ResNet-101 model, trained on the ImageNet. Results are similar to the transformer model.

Table 3: Performance of OOD detection methods based on the ResNet-101 on the ImageNet 1K, averaged over different outlier datasets. ↑ indicates that larger values are better, while ↓ indicates the opposite – all values in percent.

Outlier	Method	AUROC ↑	AUPR-IN ↑	AUPR-OUT ↑	FPR@95TPR ↓
ImageNet 1K					
	Energy	76.77	**43.99**	94.92	65.34
	MSP	70.48	39.10	93.06	69.92
Noise	MSP	70.42	37.03	92.60	71.05
Noise	Energy	78.24	43.85	94.90	63.43
Generated	MSP	77.41	42.41	94.78	63.30
Generated	Energy	**82.68**	43.52	**96.12**	**55.12**

Model Convergence We investigate how different types of outliers influence the performance of models during optimization. An overview for the CIFAR 100 is given in Fig. 4. We observe that

[2] `https://github.com/pytorch/examples/tree/main/word_language_model`
[3] `https://github.com/ajbrock/BigGAN-PyTorch`

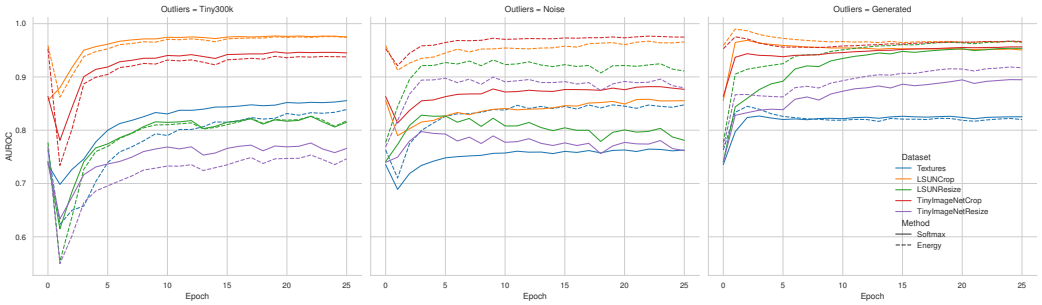
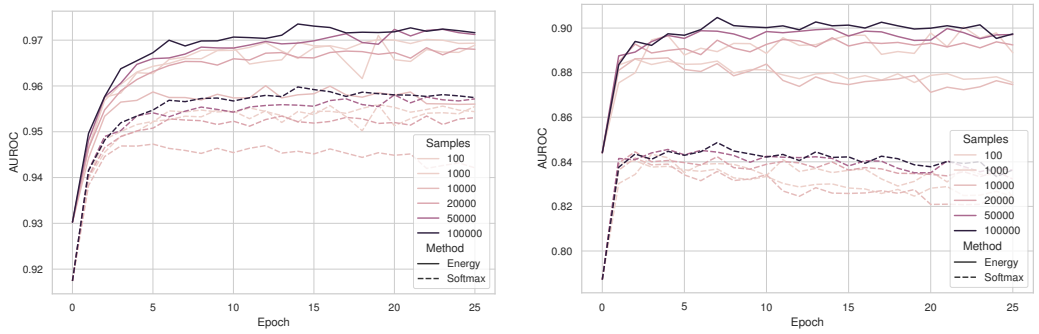


Figure 4: Performance of models during optimization on the CIFAR 100. We observe that models trained with generated outliers converge faster compared to models exposed to noise or natural outliers.



(a) CIFAR10

(b) CIFAR 100

Figure 5: Performance of MSP and Energy-Based OOD for different numbers of outliers drawn from the generative model at $\sigma^2 = 50$. We observe that larger sample sizes correlate with better performance. However, even 100 samples can increase the performance significantly.

models trained with generated outliers tend to converge faster than models exposed to noise or natural outliers. However, the effects of outliers seem to vary between datasets. Generated outliers seem to increase the performance on TinyImageNet Resize, and LSUN Resize the most. These results suggest that combining different kinds of outliers could increase performance.

Number of Outliers To investigate the influence of the number of outliers drawn from the generative model, we measured the performance of models for various outlier dataset sizes. The results are depicted in Fig. 5. Generally, a larger number of outliers is associated with higher performance at the end of the training. While this effect seems to saturate at $\approx 50,000$ samples, we see that even 100 outliers can significantly improve the performance.